

EXHIBIT B



Black Ice: The Law Enforcement Freenet Project

Wayne Becker, Special Investigator
Missouri ICAC Task Force
Dent County Sheriff's Office

Black Ice: The Law Enforcement Freenet Project

Abstract

Freenet is a distributed, Internet-wide, peer-to-peer overlay network designed to allow anonymous and censorship resistant publication and distribution of information. The system is regularly used to distribute and download child abuse material. The Black Ice Project is an effort of the Missouri ICAC Task Force to discover a way to identify subjects using *Freenet* to exchange child abuse material and develop sufficient probable cause to obtain a search warrant.

This paper describes the basic functioning of *Freenet*, how it is used by persons exchanging child abuse material, the system's vulnerabilities and how the Black Ice project exploits them. This paper also contains a section on the forensic opportunities when *Freenet* is encountered either as a result of a Black Ice investigation or from another source. An in-depth discussion of *Freenet* and its theories of operation is not in the scope of this paper and the reader is referred to the references section for sources of a more detailed and academic discussion.

While this paper will describe the technique used to obtain probable cause for a search warrant this paper alone is not enough education to use the output of Black Ice to make criminal cases. Additional training, including a hands on lab environment, is essential to a thorough understanding of the topic.

Funding for this project was made available by the Missouri Multi-Jurisdictional Cyber Crime Grant (MJCCG), which was created by funding made available by the U.S. Department of Justice, Bureau of Justice Assistance and the State of Missouri, Department of Public Safety under the American Recovery & Reinvestment Act of 2009; the Missouri ICAC Task Force through the PROTECT Act of 2008; and the Missouri State Cyber Crime Grant (SCCG) through the Missouri Department of Public Safety.

Freenet

Freenet is a peer-to-peer platform for censorship-resistant, secure, and anonymous communication and file sharing on the Internet. It was originally designed by Ian Clarke, a computer science student at the University of Edinburgh, Scotland. In Clarke's final year at Edinburgh, he completed his final year project, entitled "A Distributed, Decentralised Information Storage and Retrieval System". In July 1999 Clarke decided to release it to the Internet and invited volunteers to help implement his design. The resulting free software project became known as *Freenet*. According to Clarke, *Freenet* aims to provide freedom of speech through a peer-to-peer network with strong protection of anonymity; as part of supporting its users' freedom, *Freenet* is free and open source software. It uses decentralized, distributed data stores to store information, and has a suite of free software for working with this data store. Users contribute to the network by giving bandwidth and a portion of their hard drive (called the *datastore*) for storing files. Files are automatically kept or deleted depending on how popular they are, with the least popular being discarded to make way for newer or more popular content. Files are encrypted, so generally the user cannot easily discover what is in his data store. Chat forums, websites, and search functionality, are all built on top of this distributed data store.

Freenet works by storing small encrypted blocks of content distributed on the computers of its users and connecting only through intermediate computers which pass on requests for content and sending them back without knowing the contents of the full file, similar to how routers on the Internet route packets without knowing anything about files. Except *Freenet* provides caching, encryption, and does not rely on centralized structures. This allows users to anonymously publish or retrieve various kinds of information.

Through the use of separate applications, or plugins, loaded along with the *Freenet* software, users can interact with the network in other ways, such as forums similar to web forums or Usenet. *Frost* is one such forum application.

While *Freenet* provides an HTTP interface for browsing *Freesites*, Internet sites that only exist within *Freenet*, it is not a proxy for the World Wide Web. *Freenet* can only be used to access content that has been previously inserted into the *Freenet* network. *Freenet* attempts to protect the anonymity of both subjects inserting data into the network (uploading) and those retrieving data from the network (downloading). Unlike other file sharing systems, there is no need for the subject uploading the file to remain on the network after uploading a file or group of files. Instead, during the upload process, the files are broken into blocks, or splits, and stored on a variety of other computers on the network. When downloading, those blocks are found and reassembled.

Unlike other P2P networks, *Freenet* not only transmits data between nodes (nodes are computers running *Freenet*) but actually stores data, working as a huge distributed cache. Files on *Freenet* are split into multiple small blocks, with additional blocks added to provide redundancy. Each block is handled independently, meaning that a single file may have parts stored on many different nodes. An index to the complete file is contained in a high level block, or manifest.

A user wishing to share a file "inserts" the file to the network. After insertion is finished, the user receives the key to the file and then the uploading node is free to shut down, because the file is stored in the network. It will remain available for other users whether or not the original uploading node is online. No single node is responsible for the content; instead, it is replicated to many different nodes. However, there is no ability to search for a file key that has been inserted into the network. The user sharing the file has to make the key available to either an individual using something like email, or to a broader audience by posting it on a message board, forum or on a *Freesite*.

Freenet provides two (2) modes of operation, *OpenNet* and *DarkNet*. *OpenNet* makes connections to any other *OpenNet* node, or strangers. In *DarkNet* mode, connections are only made only to friends a user has previously exchanged node ids with, creating a closed network.

Freenet Keys

Data is referenced and retrieved using keys. *Freenet* uses three (3) main key types:

Updatable Subspace Key (USK) is primarily used for *Freesites* sites, web pages stored within *Freenet*. It has a public key to retrieve the site webpage and a private key used to update site.

Signed-Subspace Key (SSK) is similar to USK, but used for files. It has a public key used to retrieve files and a private key used to insert the file. If two (2) different users insert the identical file using an SSK each will have a unique key.

Content Hash Key (CHK) is the basic key used for static data such as images, video, and documents. It is the most common key in *Freenet* and is also the key for all the underlying splits for a file with a SSK. A CHK is a SHA256, base 64 hash of a document. The CHK is unique by nature and provides for tamperproof content. CHKs also reduce the redundancy of data since the same data will have the same CHK. The CHK consists of three parts:

1. the hash for the file
2. the decryption key that unlocks the file, and
3. the cryptographic settings used

A typical CHK key looks like this:

CHK @ file hash , decryption key , crypto settings

Or for example:

CHK @ SVbD9~HM5nzf3AX4yFCBc-A4dhNUF5DPJZLL5NX5BrS , bA7qLNJR7IXRKn6uS5PAySjIM6azPFvK~18kSi6bbNQ , AAEA--8

The full (or manifest) key is required for an end user to retrieve a file, but only the file hash portion of the key is passed through the network, so an intermediate node does not have the ability to decrypt the file. The decryption key and crypto settings are required. The crypto settings tell *Freenet*, the type of encryption used, if the file is compressed and how, and the version of *Freenet* the file was inserted under for backward compatibility.

For each file there are very many CHK keys (or CHK blocks) inserted, each is 32kb in size. The first block is the manifest that contains metadata about the file, and lists the CHKs to all split file data and check blocks. This is the block that the shared CHK/SSK key points. If this manifest disappears from the network, the download cannot start.

Split File Manifest Example:

```
Split file: CHK@RNWeZbc9pj5DFB4IFy0kYuXMOFE9E865DnmA~~
jV5O4,IzX4CgyrnxBYVZoIEy6UuAFY9Uwm4rX7UOCmKLIHP2c,AAMC--
8/Daddy%27s%20Girl%201.wmv
Split file info
Metadata version 1
Split file type: FEC Onion standard (1)
Compatibility: COMPAT_UNKNOWN (min: COMPAT_1416 max: COMPAT_1416)
Splitfile CryptoKey: 219c780a0cab9f1072559a25132e94b80158f54c26e2b5fb50e08c90b2073f67
Hashes:
SHA1: 7f9ab2a9bda27ff6700ad270fa89b18f150df259
MD5: 86d88f721f4ea2d29713dcbf76648b8d
SHA256: 4e3317d4cb12559b520ccb9180659fb9386ab3ac0db54ccdf8d6c38dc68e656e
SHA512:
c89ba88bbb7db64d71567d0f601dfd5083ee05afeab34492fff3a7cf68332983d4ea1270b2d60a5f9f9583fbfb3d39
13cc74e421dd17ed500b000fe0d186e478
```

```
ED2K: 47462801da85b34e66b93cc964369058
TTH: be4c97a0f92f17e34c80f8ffb3660ecd7e99686ea08c783e
Uncompressed data size: 6597770 bytes.
Segment count: 2
Data blocks per segment: 101
Check blocks per segment: 102

Segment #0
Data Blocks: 101
0 CHK@EAEw6BTt-
EWkVUUPcVA~vGg7RCeP~ObiKtYVKtPmQA8, IZx4CgyrnxBYVZolEy6UuAFY9Uwm4rX7UOCMkLIHP2c, AAM
A--8
... .
```

The split keys are divided into equal segments of no more than 128 keys (256 with the check block keys). If all the keys for a file will fit in the manifest then the keys point to the actual data blocks of the file. In fact, if the file is less than 32kb, the file itself is contained in the manifest block. Otherwise, the keys in the manifest point to additional blocks of equal size containing the keys to all data and check blocks. If the file is large enough to require split file blocks the split file blocks are also required to be able to download the file.

The data blocks are the actual file and it is possible for some of these to fall out of the network. Using a forward error correction technique the check blocks provide redundancy to be able to recreate missing data blocks or *heal* files. If some data blocks are missing, check blocks are used to recover the data. Check blocks themselves can also be recovered from the data blocks. This healing process results in the occasional insert of a block for a file occurring in the middle of the requests to retrieve blocks of the same file.

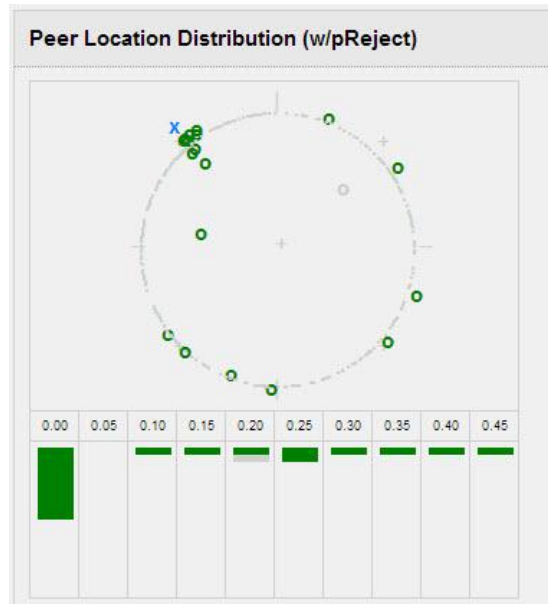
When you download a file, the manifest is first downloaded, followed by all split file blocks. After that the real download begins. *Freenet* will download data blocks and check blocks at random, until it has enough to fully reconstruct the file. The manifest key includes the file name at the end. If the same file is inserted into the network twice with different filenames it will create a different manifest key, however, if it is a CHK key all the underlying data blocks will be the same and will have the same keys.

Routing

Freenet must be able to determine which nodes to store data on, and later be able to find that data again. The process of finding a piece of data, or a place to store it, is called routing. Because nodes connect with a limited number of peers and communicate only with them, routing is very difficult because it must be done with only locally available information.

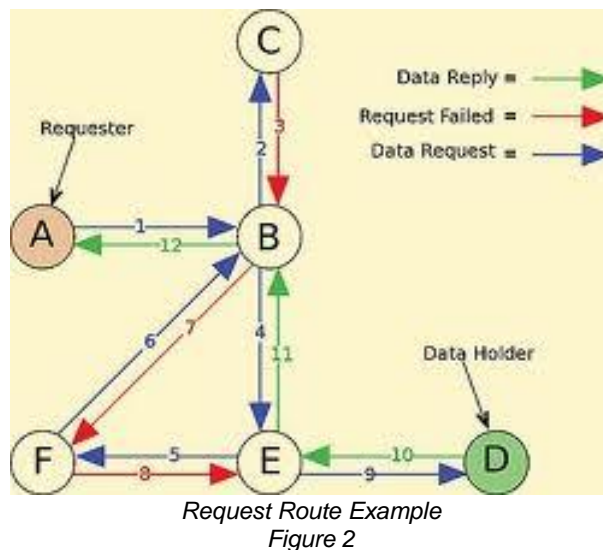
In graph theory, there is a type of network called a small-world network. A small-world network contains relatively short routes between any two nodes. Some types of small-world networks are especially useful because they allow finding short routes with only locally available information. All nodes have a location, unrelated to geographical location, which is a number in the range 0 (inclusive) to 1 (exclusive). Every request has an inherent ideal location which it is routed towards. Nodes route requests by giving them to the peer whose location is closest to that ideal location. However, in order for this to be effective, the network must have very specific characteristics.

Locations can be thought of as wrapped around a circle: zero at one point, approaching 1 as it goes around, then wrapping back to zero. In this model 0.3 is 0.2 away from 0.5, and 0.9 is 0.2 away from 0.1. This distance between peers' locations is called the connection's link length. On average, nodes must have many connections with shorter link lengths, and a few connections with longer link lengths. One can think of this as being able to quickly make large leaps on the location circle and also make small adjustments. Figure 1 is an example of how peer nodes are distributed, with the x indicating the local node and the o the location of peers connected.



Sample Peers Distribution
Figure 1

When a key is requested, first the node checks the local data store. If it's not found, the key's hash is turned into another number in the same zero to 1 range, and the request is routed to the node whose location is closest to the key. This goes on until some number of hops is exceeded, there are no more nodes to search, or the data is found (see figure 2). If the data is found, it is cached on each node along the path. So there is no one source node for a key, and attempting to find where it is currently stored will result in it being cached more widely.



In *OpenNet*, a node can connect with untrusted peers, called strangers. In an attempt to improve link length distribution, nodes perform something called path folding. Path folding can happen when a request for a block of data succeeds. The endpoint can return an offer to connect (its *OpenNet* node reference) along with the requested data. As the data travels backwards along the route to return to the node which made the request, a node along the way can accept the offered connection, and the two become peers. To protect the anonymity of the endpoint, the node which accepted the connection removes the offer to connect, and the next node on the way back can add its own. New connections are then sometimes added to downstream nodes (i.e. the node that answered the request) when requests succeed, and old nodes are discarded in least recently used order (or something close to it). Oskar Sandberg's research (during the development of the latest version) shows that this path folding is critical to the performance of the network, and that a very simple routing algorithm will suffice provided there is path folding. The disadvantage of this is that it is very easy for an attacker to find *Freenet* nodes, and connect to them, because every node is continually attempting to find new connections.

Frost

Frost is a free, open source, Java application that runs under *Freenet* and provides message boards or forums, similar to Usenet, as well as the ability to upload and download content. *Frost* is a very popular front-end for *Freenet* since it provides for bulletin boards for public messages as well as the ability to private message other users.

A user can post and reply to messages without establishing an identity within the *Frost* system, however, many established users ignore such anonymous posts. A user can establish a nickname and *Frost* will append a unique identifier. This allows multiple users to have the same base nickname and also prevents someone from pretending to be another user. When a user posts a key to a file on a board message in *Frost*, the key will be highlighted, similar to the way a hyperlink appears on a web page. The user only has to click on the key and a box opens where he can select download for the file to be added to the download queue.

Frost is commonly and blatantly used to trade child abuse material. Users can upload a file into the *Freenet* network and then share it by posting the key on a *Frost* message board. Many

message boards have names with common child pornography terms such as *PTHC*, *girls.10-13*, *boyporn*, *hussy* and *hurtcore* (see Figure 3). *PTHC*, *hussy* and *child models-girls* are consistently the most active boards on the *Frost* system.

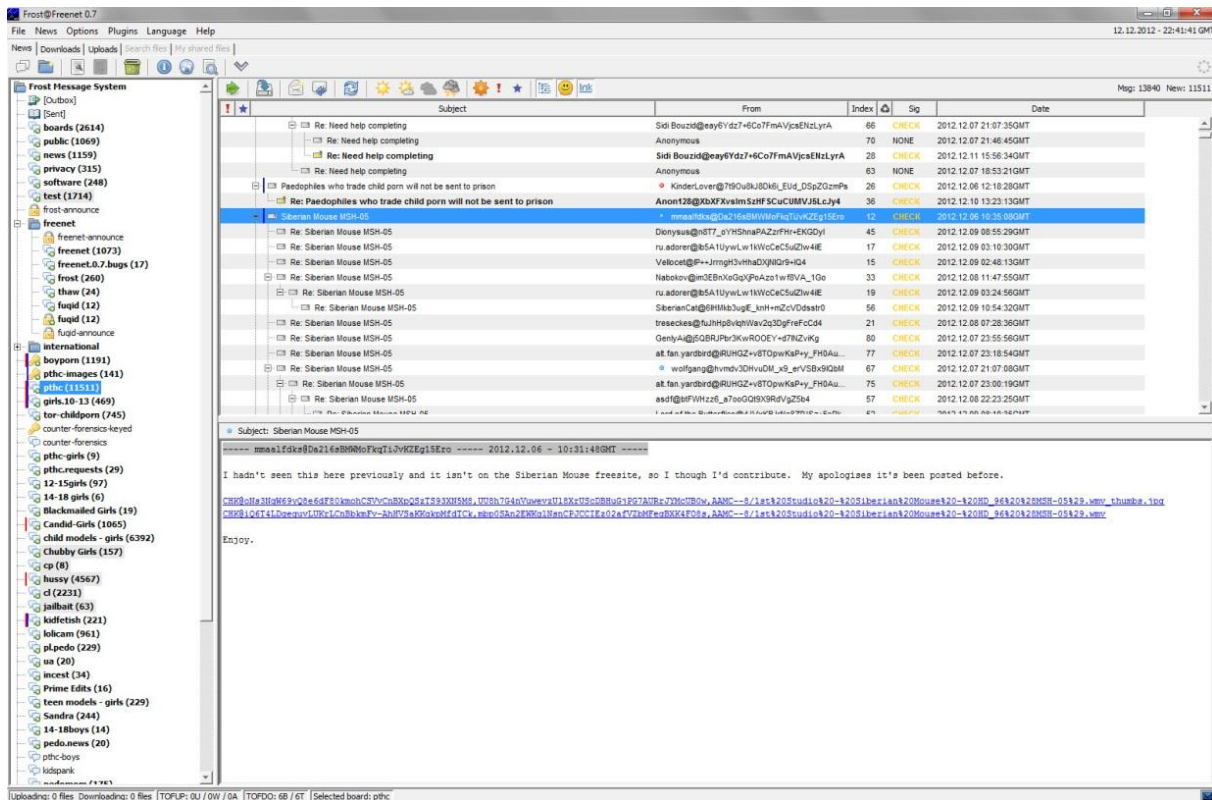


Figure 3
Frost Main Screen

Frost activity varies widely and the system is vulnerable to Denial of Service attacks. *Frost* has fallen from favor for a lot of the main stream messaging; instead users have migrated to *FMS* (Freenet Message System). *FMS*, however, is generally hostile to child pornography traders so *Frost* remains the messaging system of choice for subjects trading child abuse materials.

In addition to a source of manifest keys to files, many posts provide instructions on how to avoid getting caught by the authorities, groom children, or encrypt or hide your collection. There was even a post of all the current ICAC commanders and their email addresses with the suggestion other users should anonymously send child pornography pictures to their email boxes.

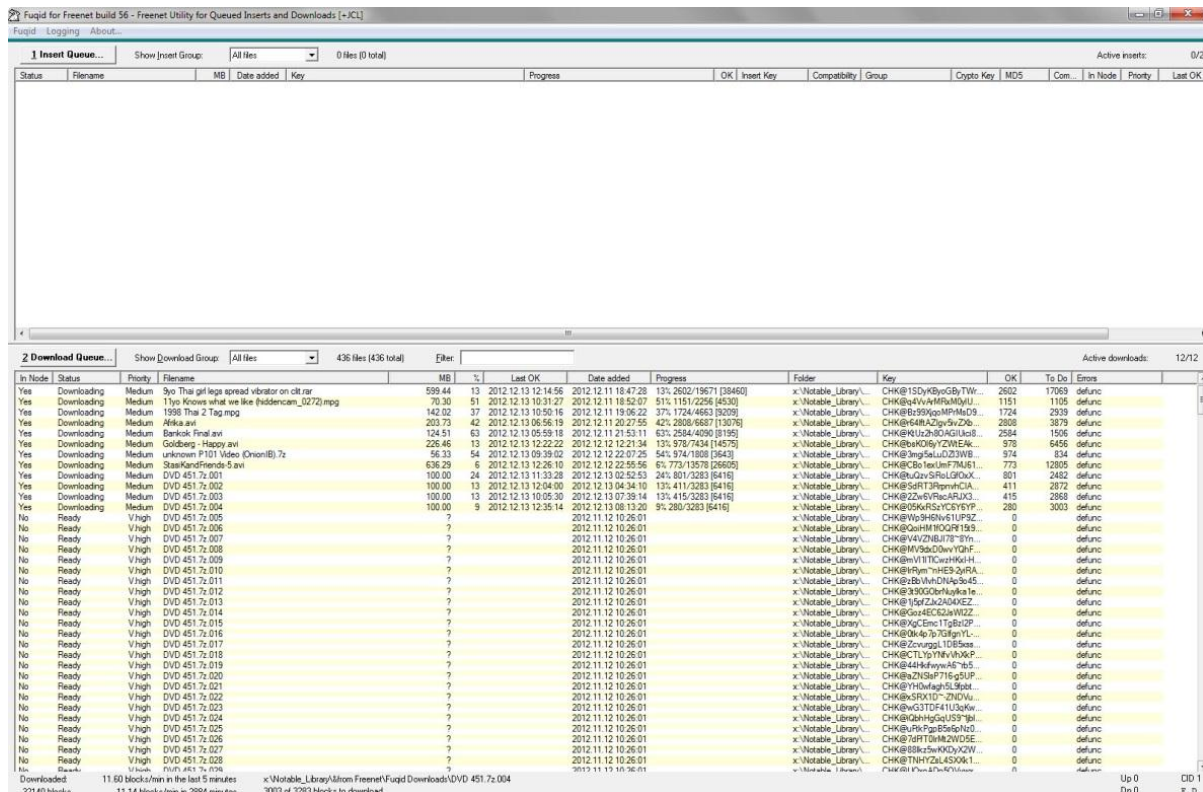
Fuqid

Fuqid is an external application for *Freenet* that is used as an insert/download manager for files. *Fuqid* stands for: *Freenet* Utility for Queued Inserts and Downloads. It is written under Delphi 7 and runs natively on Windows or on Linux under Wine.

Fuqid offers a user significant control over both inserts and downloads of files into *Freenet* (see figure 4). It can be used as an independent file manager or in conjunction with *Frost*. When selecting a key in *Frost* a user is given the option of copying the key to the clip board (as well as downloading). A user only has to right click within the *Fuqid* window and select *add* to put the

Black Ice: The Law Enforcement Freenet Project

file in the download queue. *Fuqid* keeps track of your previous downloads and lets the user know when the file already exists.



The screenshot displays the Fuqid application window, titled "Fuqid for Freenet build 56 - Freenet Utility for Queued Inserts and Downloads [-XCL]". It features two main sections: "1 Insert Queue..." and "2 Download Queue...".

1 Insert Queue...

Status	Filename	MB	Date added	Key	Progress	OK	Insert Key	Compatibility	Group	Crypto Key	MOS	Com...	In Node	Priority	Last OK
Yes	Download	Medium	599.44	13	2012 12 13 12 14 56	2012 12 11 18 47 28	131 2602/19671 [38460]	x\Notable_Library\	CHK@150yK9yGByTW...	2602	17069	defunc			
Yes	Download	Medium	70.30	51	2012 12 13 10 31 27	2012 12 11 18 52 07	511 1151/2256 [6530]	x\Notable_Library\	CHK@4V4v4v4v4v4v...	1151	1105	defunc			
Yes	Download	Medium	142.02	37	2012 12 13 10 50 16	2012 12 11 19 06 22	371 1724/4663 [30209]	x\Notable_Library\	CHK@6z599z9z9z9z...	1724	2939	defunc			
Yes	Download	Medium	203.73	42	2012 12 13 06 56 19	2012 12 11 20 27 55	421 2809/6687 [13076]	x\Notable_Library\	CHK@644R4z9z9z9z...	2808	3879	defunc			
Yes	Download	Medium	124.51	63	2012 12 13 05 59 19	2012 12 11 21 53 11	631 2554/4090 [9195]	x\Notable_Library\	CHK@W42z9z9z9z9z...	2554	1505	defunc			
Yes	Download	Medium	226.46	13	2012 12 13 12 22 22	2012 12 12 12 21 34	131 978/7434 [14575]	x\Notable_Library\	CHK@8aX0D6yZWZAK...	978	6456	defunc			
Yes	Download	Medium	56.33	54	2012 12 13 09 09 02	2012 12 12 22 07 25	541 574/1808 [3643]	x\Notable_Library\	CHK@3mg5wLwD23WB...	574	834	defunc			
Yes	Download	Medium	636.29	6	2012 12 13 12 26 10	2012 12 12 22 55 56	61 773/13578 [26605]	x\Notable_Library\	CHK@Bo4w4w4w4w4w...	773	12005	defunc			
Yes	Download	Medium	100.00	24	2012 12 13 11 33 28	2012 12 12 02 52 53	241 801/3283 [8416]	x\Notable_Library\	CHK@ZuZySRuLGOx...	801	2482	defunc			
Yes	Download	Medium	100.00	13	2012 12 13 12 04 08	2012 12 13 04 34 10	131 411/3283 [8416]	x\Notable_Library\	CHK@5dRT3RpmHCA...	411	2872	defunc			
Yes	Download	Medium	100.00	13	2012 12 13 10 05 39	2012 12 13 07 39 14	131 415/3283 [8416]	x\Notable_Library\	CHK@ZuZySRuLGOx...	415	2868	defunc			
Yes	Download	Medium	100.00	9	2012 12 13 12 35 14	2012 12 13 08 13 20	91 280/3283 [8416]	x\Notable_Library\	CHK@09K6rSsYCY6Y...	280	3003	defunc			
No	Ready	V/High	DVD 451 7z 005	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Wp3Hw451UP3Z...	0	defunc				
No	Ready	V/High	DVD 451 7z 006	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Wp3Hw451UP3Z...	0	defunc				
No	Ready	V/High	DVD 451 7z 007	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@4V4ZNBu787Yn...	0	defunc				
No	Ready	V/High	DVD 451 7z 008	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Mv99wCwvYQF...	0	defunc				
No	Ready	V/High	DVD 451 7z 009	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Wp3Hw451UP3Z...	0	defunc				
No	Ready	V/High	DVD 451 7z 010	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Pym7mE3ZyRA...	0	defunc				
No	Ready	V/High	DVD 451 7z 011	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Bv4hNDHAgsc45...	0	defunc				
No	Ready	V/High	DVD 451 7z 012	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@3X0CzRkUk1e...	0	defunc				
No	Ready	V/High	DVD 451 7z 013	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@15ofZa2A04vEZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 014	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 015	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 016	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 017	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 018	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 019	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 020	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 021	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 022	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 023	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 024	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 025	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 026	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 027	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 028	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				
No	Ready	V/High	DVD 451 7z 029	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc				

2 Download Queue...

In Node	Status	Priority	Filename	MB	%	Last OK	Date added	Progress	Folder	Key	OK	To Do	Errors
Yes	Download	Medium	599.44	13	2012 12 13 12 14 56	2012 12 11 18 47 28	131 2602/19671 [38460]	x\Notable_Library\	CHK@150yK9yGByTW...	2602	17069	defunc	
Yes	Download	Medium	70.30	51	2012 12 13 10 31 27	2012 12 11 18 52 07	511 1151/2256 [6530]	x\Notable_Library\	CHK@4V4v4v4v4v4v...	1151	1105	defunc	
Yes	Download	Medium	142.02	37	2012 12 13 10 50 16	2012 12 11 19 06 22	371 1724/4663 [30209]	x\Notable_Library\	CHK@6z599z9z9z9z...	1724	2939	defunc	
Yes	Download	Medium	203.73	42	2012 12 13 06 56 19	2012 12 11 20 27 55	421 2809/6687 [13076]	x\Notable_Library\	CHK@644R4z9z9z9z...	2808	3879	defunc	
Yes	Download	Medium	124.51	63	2012 12 13 05 59 19	2012 12 11 21 53 11	631 2554/4090 [9195]	x\Notable_Library\	CHK@W42z9z9z9z9z...	2554	1505	defunc	
Yes	Download	Medium	226.46	13	2012 12 13 12 22 22	2012 12 12 12 21 34	131 978/7434 [14575]	x\Notable_Library\	CHK@8aX0D6yZWZAK...	978	6456	defunc	
Yes	Download	Medium	56.33	54	2012 12 13 09 09 02	2012 12 12 22 07 25	541 574/1808 [3643]	x\Notable_Library\	CHK@3mg5wLwD23WB...	574	834	defunc	
Yes	Download	Medium	636.29	6	2012 12 13 12 26 10	2012 12 12 22 55 56	61 773/13578 [26605]	x\Notable_Library\	CHK@Bo4w4w4w4w4w...	773	12005	defunc	
Yes	Download	Medium	100.00	24	2012 12 13 11 33 28	2012 12 12 02 52 53	241 801/3283 [8416]	x\Notable_Library\	CHK@ZuZySRuLGOx...	801	2482	defunc	
Yes	Download	Medium	100.00	13	2012 12 13 12 04 08	2012 12 13 04 34 10	131 411/3283 [8416]	x\Notable_Library\	CHK@5dRT3RpmHCA...	411	2872	defunc	
Yes	Download	Medium	100.00	13	2012 12 13 10 05 39	2012 12 13 07 39 14	131 415/3283 [8416]	x\Notable_Library\	CHK@ZuZySRuLGOx...	415	2868	defunc	
Yes	Download	Medium	100.00	9	2012 12 13 12 35 14	2012 12 13 08 13 20	91 280/3283 [8416]	x\Notable_Library\	CHK@09K6rSsYCY6Y...	280	3003	defunc	
No	Ready	V/High	DVD 451 7z 005	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Wp3Hw451UP3Z...	0	defunc		
No	Ready	V/High	DVD 451 7z 006	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Wp3Hw451UP3Z...	0	defunc		
No	Ready	V/High	DVD 451 7z 007	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@4V4ZNBu787Yn...	0	defunc		
No	Ready	V/High	DVD 451 7z 008	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Mv99wCwvYQF...	0	defunc		
No	Ready	V/High	DVD 451 7z 009	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Wp3Hw451UP3Z...	0	defunc		
No	Ready	V/High	DVD 451 7z 010	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Pym7mE3ZyRA...	0	defunc		
No	Ready	V/High	DVD 451 7z 011	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Bv4hNDHAgsc45...	0	defunc		
No	Ready	V/High	DVD 451 7z 012	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@3X0CzRkUk1e...	0	defunc		
No	Ready	V/High	DVD 451 7z 013	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@15ofZa2A04vEZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 014	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 015	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 016	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 017	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 018	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 019	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 020	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 021	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 022	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 023	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 024	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 025	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 026	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 027	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 028	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		
No	Ready	V/High	DVD 451 7z 029	?	?	2012 11 12 10 26 01	?	x\Notable_Library\	CHK@Qz45C32wWQZ...	0	defunc		

Downloaded: 11.60 blocks/min in the last 5 minutes
32140 blocks
11.14 blocks/min in 2884 minutes
3003 of 3283 blocks to download

Active inserts: 0/2
Active downloads: 12/12

Figure 4
Fuqid Upload/Download Screen

The Black Ice Project

During the initial research into *Freenet* we discovered it was easy to identify IP addresses that were using the program and had connected to our node. (note: the Black Ice project only connects to the *OpenNet*).

Using *Freenet* in the lab we discovered that downloading a file can be extremely slow and requires much patience, as well as the requirement to leave the system running almost 24/7. At least in the United States, we saw little legitimate use for *Freenet* beyond foreign nationals and maybe academic research. After a few knock and talk attempts we also quickly recognized that subjects using *Freenet* were more technically sophisticated and aware of their right to deny us consent to search.

The objective of the Black Ice Project was to find a way to establish probable cause that a *Freenet* user is retrieving child abuse material from the network. Since the data is encrypted and distributed across many nodes, unlike other P2P networks, we cannot directly download a file from a subject. However, we can "see" what a node is requesting from the network if it is directly connected to us.

The current developer (and the only one paid) is Matthew Toseland, a college student in the UK. He goes by the handle *Toad* and he frequently posts opinions, directions and future improvements to *Freenet* on the developer web site, IRC or his *Flog* (*Freenetese* for a blog). He recently discussed some of the vulnerabilities in *Freenet*:

Correlation attack: If you are very close to the originator, and can identify a stream of requests, you may be able to identify it simply by the fact that there is a large proportion of the known download coming from a peer. This works a few hops away too IMHO. So it's both malicious peers and attackers controlling an (as yet unquantified) proportion of the network.

... Freenet is about hiding in the crowd. It's not much use if you are already on their shortlist of suspects. In the more unpleasant regimes, and even in the west quite often, you have bigger problems at this point (e.g. dawn raids!). In particular, Freenet does not provide strong protection if you are already connected to the target: Provided you can recognize the content (possibly after the fact from logs), you can identify with reasonable confidence that they are the originator. One consequence of this is you need a reasonably large network to hide in, and if the bad guy can afford to connect to everyone (or in some cases exclude a group of nodes at a time) he can probably trace you.

We do not provide the sort of anonymity guarantees that mixnets such as Tor can in theory provide...

In Ian Clarke's paper, *Private Communication Through a Network of Trusted Connections: The Dark Freenet*, where DarkNet is introduced, he states:

A second concern, that has come to the forefront with the actual deployment of Freenet, is the vulnerability of people operating nodes in the network. While the network strove to dissociate users from the data they accessed and nodes from the data they served, it did not hide that a particular node was part of the network. In order to find Freenet nodes in earlier incarnations of the system, it was sufficient to join the network and start operating as a node oneself: through the continuous process of routing and optimizing the network, one would eventually learn the identities and Internet addresses of more and more nodes in the network. This means that somebody wishing to attack the Freenet network in its entirety would have had no problem finding and identifying participants given sufficient time.

This vulnerability is basically what the Black Ice project is exploiting by identifying nodes that are retrieving content that is known or suspected to be child abuse material. Since 2011, we have been collecting the high level manifest keys from posts on *Frost* and other *Freesites*. In the spring of 2012, programing was done by a Missouri ICAC Investigator, Wayne Becker, at the Salem lab to automate the retrieval of the split keys associated with the collected manifest keys. These keys are maintained in a SQLite database. As of August 1, 2013 we have approximately 20,000 manifest keys, or files of interest, with over 40 million split keys in the Black Ice keymaster SQL data base. The average file of interest has approximately 2,200 split keys associated with it, making it approximately 36MB in size. This data base continues to grow as

we find new manifest keys to add. It is worth noting that images are typically distributed in *rar* archives and when these files are extracted we have a database of over 50,000 file hashes.

Freenet is an open source project and is in constant development. The system is written in Java and the source code is readily available. In April of 2012, a Research Scientist with the University of Massachusetts Amherst, Dr. Marc Liberatore, modified the *Freenet* program for the Black Ice Project so that it logs the IP address, CHK value, HTL (hops to live), type of request, and date/time as a request passes through the node. This modified version of *Freenet* (known as *LEFn*) has been initially deployed at six (6) law enforcement locations in Missouri. The data collected by these *LEFn* nodes is forwarded nightly to the lab in Salem, MO. Since we are only aware of the IP addresses directly connected to a *LEFn* node, the more connections to "strangers" increases the number of suspect IP addresses located.

Black Ice essentially mines data from these log records. Programs developed at the Salem lab do two (2) basic functions; they look for any IP address that is using *Freenet*, and look for specific records that contain requests for part of a file known or suspected to be child abuse material (file of interest). This results in the maintenance of two (2) SQLite databases. The first contains any unique, IP address/port combination ever seen by a *LEFn* node, and with the first and last date it was observed. This database can be used as an investigative tool to determine if an IP address seen elsewhere is also using *Freenet*. With the current deployment of *LEFn* nodes we are seeing 3,000-4,000 new US IP address/port combinations each month and approximately 15,000 globally.

The port number is of interest because *Freenet* randomly assigns a UDP port number when the system is installed. While this number is not unique across the network, it can be used to track a subject within a narrow geographic area as his IP address changes. We have seen a number of cases, including one of our own nodes, where the port number appears to change on the fly. This appears to be the result of a NAT firewall changing the outbound port number.

The second database contains specific information when the key in a logged record matches a file of interest in the KeyMaster database. Matches located in the US and Canada, and with an HTL value of 18, 17 or 16, are added to a suspect data base along with specific information about the associated file (i.e. manifest key and SHA values).

Freenet uses a value known as hops-to-live (HTL) to control how far downstream from the requester an attempt to retrieve a file block should travel. The default HTL value in *Freenet* is 18. This value is forwarded to the next node down the line and decremented as the request travels down a path. Since the only IP addresses known to a *LEFn* node are the ones directly connected to it, there is currently no way to tell what IP requested a key if the HTL is less than 18 (even though Toad alludes to the possibility in his posts). There is also the possibility that a request with a HTL of 18 is not from the originator but randomly (50/50 chance) passed along without being decremented. A feature, called probabilistic HTL, will change the HTL when at 18, or not decrement it, as it routes the request. This is intended to make it more difficult to establish, absolutely, the node that made the request. Our testing and observations show that the originating node does not decrement the HTL, so any HTL less than 18 is not from the original requestor.

It is possible to change the max or starting HTL within the configuration screens for *Freenet*. There is a caution that this field is for developmental use only should not be changed. We currently only look for records assuming the default is being used.

Probable Cause

The developers of *Freenet* seem to be primarily concerned that a user have *plausible deniability* for any data on, or routed by their node. They do not seem to be concerned that a user is *probably* using *Freenet* for criminal activity as much as they are an attacker's ability to prove that they are, absolutely, with data collected from *Freenet*.

To mitigate the effect of probabilistic HTL as we analyze the suspect IP addresses, we look for multiple requests for blocks of a file of interest from a specific IP, along with the absence of HTL 16/17 records for the same file, during the same time period. If we see a 17 or 16 record mixed in with HTL 18s for the same file it means HTL 18s are the result of randomly not decrementing the HTL and the subject IP is not the requestor (figure 5).

Date/Time	IP Address	Port	CC	RC	City	File Name	HTL	Type	SHA1 (base 32)	
2013 07 01 09:29:32-0500	72.172.199.129	31333	US	MO	Salem	Daughter%27s Pussy.mpg	18	R	2N3MCQZJXE4R3I6RKWMFBNJRXMYFUTL	CHK@0X4cwHY5gYivkQwq8EWCHh2RBi91jlr8/Daughter%27s%20Pussy.mpg
2013 07 01 15:33:50-0500	72.172.199.129	31333	US	MO	Salem	alt.fan.prettyboy.7z	18	R	TUOB46MPLAJAXCJBKJZKBQ7ICIMRV13	CHK@0nwJDvPLSuaDPCmvhOwtTuPBuUta~fW
2013 07 02 01:33:03-0500	72.172.199.129	1781	US	MO	Salem	10yo 11yo 12yo girl 15yo boy1 Brazi 40m00s.mpg	18	R	VOJAL4FERKNNORD3MNB6TM4HYQWWPW4I	CHK@1AO2bjNkR61TPS0s19HVqiYX1SmFkC8/10yo%2011yo%2012yo%20girl%2015yo%20bo
2013 07 03 11:15:18-0500	72.172.199.129	2089	US	MO	Salem	10yo 11yo 12yo girl 15yo boy1 Brazi 40m00s.mpg	17	R	VOJAL4FERKNNORD3MNB6TM4HYQWWPW4I	CHK@1AO2bjNkR61TPS0s19HVqiYX1SmFkC8/10yo%2011yo%2012yo%20girl%2015yo%20bo
2013 07 03 13:45:17-0500	72.172.199.129	2089	US	MO	Salem	10yo 11yo 12yo girl 15yo boy1 Brazi 40m00s.mpg	17	R	VOJAL4FERKNNORD3MNB6TM4HYQWWPW4I	CHK@1AO2bjNkR61TPS0s19HVqiYX1SmFkC8/10yo%2011yo%2012yo%20girl%2015yo%20bo
2013 07 02 02:52:59-0500	72.172.199.129	1781	US	MO	Salem	Klka sex.avi	16	R	M4C7MTKO4PLQW7BGFNQ3SZQGOGZNXZLM	CHK@1EVtEDHPOzLzw5F718jwGKbAKVdYTv
2013 07 03 10:29:35-0500	72.172.199.129	31333	US	MO	Salem	Klka sex.avi	18	R	M4C7MTKO4PLQW7BGFNQ3SZQGOGZNXZLM	CHK@1EVtEDHPOzLzw5F718jwGKbAKVdYTv
2013 07 03 10:49:40-0500	72.172.199.129	31333	US	MO	Salem	Klka sex.avi	16	R	M4C7MTKO4PLQW7BGFNQ3SZQGOGZNXZLM	CHK@1EVtEDHPOzLzw5F718jwGKbAKVdYTv
2013 07 03 13:04:11-0500	72.172.199.129	2089	US	MO	Salem	Youngvideomodels - JIL01 - Julia 10yo, Ira 13yo & Luda 14yo.avi	16	R	JS4YVLQYGXAUAAOBL5WCOWD73PGXXI6	CHK@1HUBH0p13DXn-ixu0S0q7rMKryFQQgK8/Youngvideomodels%20-%20JIL01%20-%20Juli
2013 07 03 15:04:26-0500	72.172.199.129	2089	US	MO	Salem	Youngvideomodels - JIL01 - Julia 10yo, Ira 13yo & Luda 14yo.avi	18	R	JS4YVLQYGXAUAAOBL5WCOWD73PGXXI6	CHK@1HUBH0p13DXn-ixu0S0q7rMKryFQQgK8/Youngvideomodels%20-%20JIL01%20-%20Juli
2013 07 03 15:05:54-0500	72.172.199.129	2089	US	MO	Salem	Youngvideomodels - JIL01 - Julia 10yo, Ira 13yo & Luda 14yo.avi	18	I	JS4YVLQYGXAUAAOBL5WCOWD73PGXXI6	CHK@1HUBH0p13DXn-ixu0S0q7rMKryFQQgK8/Youngvideomodels%20-%20JIL01%20-%20Juli
2013 07 02 02:53:22-0500	72.172.199.129	1781	US	MO	Salem	Lollipop-Issue_3-03.avi	18	R	4GJNMRBC7BU2MQGZ5KFOL5ALPBABL277	CHK@1W4DoNj9Af6h1ID8ByPvVKY61K7H0
2013 07 02 18:58:57-0500	72.172.199.129	31333	US	MO	Salem	1st-Studio Siberian Mouse HD_125 (M-11).wmv	18	R	ELQPQFCFNGHUUZGZXGYXQMJHLHO5QAF	CHK@1yap0dquX3jakg2hM3pUkzgeMIQe6dxxKStudio%20Siberian%20Mouse%20HD_125%20%
2013 07 02 19:22:39-0500	72.172.199.129	31333	US	MO	Salem	Lolhouse-SA-19.avi	18	R	V7ICLZ64ZIRL6QNJDIRNQDOUKUQGKSWI	CHK@23VlzxGJv1XjHxN6~ra7IdDgMjMXTf
2013 07 03 11:22:51-0500	72.172.199.129	2089	US	MO	Salem	hc_155_1.mpg	17	R	TINCRJKHBBGLRPWNJ3LEPYIWWVEKSIZ	CHK@2JM~b39JU3qA7HutVFKTP9c71rSAL3E
2013 07 03 16:21:22-0500	72.172.199.129	2089	US	MO	Salem	hc_155_1.mpg	18	R	TINCRJKHBBGLRPWNJ3LEPYIWWVEKSIZ	CHK@2JM~b39JU3qA7HutVFKTP9c71rSAL3E

Figure 5

In the above example the highlighted files are seen with a HTL of 16 or 17 in addition to a HTL of 18, indicating the HTL 18 record is the result of probalistic HTL and this IP is not the original requester.

Conversely, a number of HTL 18s for a file without any HTL 16/17s (or only ones in a different timeframe) makes it highly likely that our subject is the requestor (figure 6). In this case we can state that the IP addresses logged are for IPs directly connected to a law enforcement *Freenet* node (LEFn) and are 1) known to be running *Freenet*, 2) known to have routed a request for a file of interest, and 3) *probably* originated the request. Since the decision to decrement each routed record is a “flip of the coin” the probability of two just HTL 18 records being sent from someone other than the requestor is only 25%. Just three HTL 18 records would give us a better than 87% probability that the IP address is the requesting node.

It is not uncommon to see a file of interest with all HTL 18s records but another file of interest, from the same IP, having a mix of 18s and 16/17s. This would simply mean the suspect node was both requesting a file of interest and routing requests for a different file of interest in the same time frame. There is also the possibility that a subject is both requesting a file and routing

requests from another node for the same file in the same timeframe. Currently we have no way to determine that is what is happening and must assume the node is not the requester.

2013 07 01 13:39:32-0500	72.172.199.129	31333	US	MO	Salem	Private Photo Collection (the GL series).rar	16	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:48:04-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:49:10-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:49:28-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:49:51-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:50:27-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:51:04-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:51:05-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:51:25-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:51:35-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	I	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:51:40-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:51:46-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:52:00-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:52:15-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:52:57-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:53:19-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%
2013 07 03 16:53:37-0500	72.172.199.129	2089	US	MO	Salem	Private Photo Collection (the GL series).rar	18	R	2SOKUFMVI6QNK6Z3KBTf6YB6LDSZEIXL	CHK@Tv08KwTq8pOypVtrL8NJ2L2szd9eD9a8/Private%20Photo%20Collection%20%28the%

Figure 6

In the above example all of the records for this file have an HTL of 18. This would be an indication that the IP is the requestor and a viable target. Note the single HTL 16 record from 2 days prior. The difference in the time frames indicates this is a routed record from a different request.

Forensic Opportunities

Whether as a result of a Black Ice investigation or some other case, an examiner may come across *Freenet*, *Frost* and/or *Fuqid* in the course of an examination. Each of these programs provides its own possibilities for obtaining evidence. While this discussion focuses on Windows, *Freenet* runs quite well in a Linux or iOS environment and most of the forensics remains the same. This discussion is based on *Freenet* build 1440; *Frost* release 3302 and *Fuqid* build 56.

Freenet

Freenet is a Java based system and as such is not installed with registry entries. The default location in Windows for the current installer is *User/username/AppData/Local/Freenet*. Older installations defaulted to the users *Documents* or *My Documents* folder. However, any location can be selected at install time.

The *Freenet* folder contains a plain text *ini* file named *Freenet.ini*. This file contains all of the configuration options that a user can select as well as a few that the system sets. The main ones of interest are:

Node.downloadsDir=, the location for *Freenet* downloads. The default is ***FreenetDownloads*** but the user can specify anywhere he wants. This is where downloaded files will be placed if there is no other download manager, such as *Frost* or *Fugid*.

```
completed.list.downloads
```

The *uploads* file may contain files uploaded, preceded by the source. For example, if the file was uploaded via *Frost* the entry would be *Frost-filename*. The key of the file is not included in this log. Also, downloads made through a file manager, such as *Frost* or *Fuqid*, may not be listed in this log. This list corresponds to a list displayed on the *Freenet* web interface. If the user deletes the entries via an option on the web page, they are removed from these log files (see figure 7). The user may also select a security option that requires a password to view the file sharing pages. This option also turns off the logging of uploads and downloads.



You will likely find a *logs* folder under the *Freenet* folder, unless the user has moved logging to another location, in which case the location can be found in the *ini* file. This folder will contain the current log, the previous log and a number of compressed, older log files. These logs are for program diagnostics and run time errors. They typically do not contain anything of particular evidentiary value when the default logging value of normal is selected, but if the more verbose level of minor or debug is on the logs may contain upload and download details. The user can turn off logging completely.

Frost

Frost is also a Java application and does not use Windows registry entries. The install download is a zip file with a folder named *frost* and a number of sub-folders. No installation process is needed, so the *frost* folder could be placed anywhere. The options are found in an *ini* file, *frost.ini*, located in the folder */frost/config*. The primary options of interest are:

downloadDirectory=; this option specifies the location of the folder for file downloads. Another option, ***useBoardnameDownloadSubfolder=***, which is off by default, will cause *Frost* to create subfolders using the *Frost* board name.

logDownloads=; true by default, it controls whether *Frost* logs downloads.

logUploads=; false by default, it controls whether *Frost* logs uploads.

userName=test@Q7jC7Pn4tYlvCPTHGtDtbBdEY3o

userName.pthc=test@Q7jC7Pn4tYlvCPTHGtDtbBdEY3o

userName.test=test@Q7jC7Pn4tYlvCPTHGtDtbBdEY3o

The user name fields are created if the user creates a unique user id for posting. These ids can be different for each board the subject has posted. Their presence tells you where the subject has been posting messages. The examiner can actually use *Frost* to search for the unique user name and discover what the subject posted. This can be particularly useful if the posts are keys to contraband material. These ids can also be found in an *xml* file; *localIdentitiesBackup.xml* in a *Frost* sub-folder, *localdata*.

The *localdata* folder is also where you can find the *Frost-Downloads.log*, if download logging was left on. The download log will contain the full key of the file, followed by the filename (see figure 8). The upload log will also be in this folder if the user turned upload logging to true.


```

1 CHK@qstiJBIDxOCwuN5aW5XA8PwoMp2qmRLHjN0Si~Dt0ds,I~cYIFoQDaV~rG60NHACMyLptuGzoTweZiNrMycAE8,AAIC--8/isabel_crazy_things_with_maria.wmv_thumbs.jpg/isabel
2 crazy_things_with_maria.wmv_thumbs.jpg
3 CHK@h4mdzf7F6fGLnnYF~FR8eNq7yxFlNv18ulBVFcrrzM,2AWcoOSadQYhfrBD~ZzCgyPtwJfQmF~CpLVWYmp1daQ,AAIC--8/Isabel_Naughty_Sex_With_Albeiro.wmv_thumbs.jpg/Isabe
4 l_Naughty_Sex_With_Albeiro.wmv_thumbs.jpg
5 CHK@6EX1LyjO2XiCdN8603fATATAb17kIdcQX7DOOCDIYso,ElqMacUdrbU0Dv1~mN0WHZkKbwdvJohJZwULo2zr473E,AAIC--8/isabel_Once_Again_With_My_Friend_Lorena.wmv_thumbs.j
6 pg/isabel_Once_Again_With_My_Friend_Lorena.wmv_thumbs.jpg
7 CHK@cY~726dhzsuTDznm31W33X72YJCUUyEDEq04jepok,x2BiHYbnyFX23bg1xmE~hSUEPRxMrTG7yeyql7vJa~4,AAIC--8/isabel_my_adventures_at_the_garden.wmv_thumbs.jpg/is
8 abel_my_adventures_at_the_garden.wmv_thumbs.jpg
9 CHK@Rk1JKzey4hVDQ8c2n5FEFizH5tsqO8Q4836p9MM~iE,hdctHHb1O0dAdnZz~tgJaN2f1WXd7I~sqJbFRa10~kM,AAIC--8/Isabel_Playing_Toys_With_Marta.wmv_thumbs.jpg/Isabel
10 Playing_Toys_With_Marta.wmv_thumbs.jpg
11 CHK@rgEBTih4j1mYnc5ASemwk29blyqeftEtsD5Og~CJUQ,GQwH0~M~9Kd25~ZH3EqmgQNG19gkv5Y1~ArFW7RSWS0,AAIC--8/isabel_trio.wmv_thumbs.jpg/isabel_trio.wmv_thumbs.jp
12 g
13 SSK@5igLNGtAFuzC7q5cJS3Q1oNpPa66ICUQ8eY6B~tr1vI,KITffDL7kmRAIE3JXuq7gxsYNoFQBpRIBvtBf3TYDcA,AQACAAE/471117.jpg/471117.jpg
14 CHK@iR9qSE7isHLVT4vqqt7P6G~gNS22R2gGVjseYAKFE,qSbQvNivyoF9I6arfV4BSzHxNUAHIXqOp6XGCIhbkI,AAIC--8/Lapjes2.jpg/Lapjes2.jpg
15 CHK@fTnRJmn~uubTgoHAFqnc~eQeeMFqY1ra9bYrZXG4TJ1,5P5~JUmL8KWDwLOcql1bT3LQ0ykS9M41WQ~wHulg9HQ,AAIC--8/Charlene%20with%20PIF%20cap%201.jpg/Charlene with
16 PIF cap 1.jpg
17 CHK@gWkfth~WmdHQRTOfgmWhASgTQasW~A2B~0~Mq78Ug8,5nSdaFAFPtFpUhdH~VJ0VUCW2pLzKc6F~3opVx3gxk,AAIC--8/girl.jpg/girl.jpg
18 CHK@6AU10i1ReR2kb2xYnFGbTUSXTt210Lm3pGpXHpODk,hcAaP1CU9I7lcpFZPvGp5~ERRu5FF83~xlu2TuKQcMY,AAIC--8/ElitePain_~Cassie_Gotto.wmv/ElitePain_~Cassie_Gotto
19 o.wmv
20 CHK@xc36bf7vqk1gOrpL2L1xvZ8UdfledHYChc1R9xkTI,eqDT4mb8azjC9Hcziz808~7XXwa6GWZQ2JWH~341zUc,AAIC--8/be1c35d6752ff1e5284c9a0bfda55031.jpg/be1c35d6752ff1e
21 5284c9a0bfda55031.jpg
22 CHK@2aNUaUfLVhBbMrVEDjYZjLp7yi7c8EG1zA1req6go,zf57QSaBidUrTjQkjainz0seakG4B~D1RD~onxgzTE,AAIC--8/luckydip_38p.rar/luckydip_38p.rar
23
24

```

Figure 8
Frost Download Log

Fuqid

Fuqid is a Windows application that is distributed as an exe file. It is also purported to run in a Linux environment using Wine. As of this writing, there doesn't seem to be installation files easily available outside of *Freenet* itself. *Fuqid* comes as a zip file that extracts to a single folder named *Fuqid*. Similar to *Frost*, this folder could be placed anywhere. *Fuqid* also uses an *ini* file to contain its options. The *ini* file contains a couple of interesting artifacts:

SaveDir=; this option specifies the location where completed downloads are stored.

Keyslog=; this is the location and file name of the log file that contains a list of the keys to the files that have been inserted into *Freenet*. Logging to this file is on by default and the default file name is *Fuquids-Keys.log* in the *Fuqid* folder.

Downloadslog=; also on by default, is the location and filename of the log file that contains a list of the keys and file names that have completed downloading. It is used by *Fuqid* to avoid downloading files multiple times. Its default name is *Fuqid-Downloads.log*, also in the *Fuqid* folder.

The *ini* file also contains the complete upload and downloads queues. This is a list of in-process files and includes their keys, names and much of the metadata that is stored in the manifest block, including the various hash values of the file. This data is particularly useful evidence of receipt and distribution if the files are contraband material.

There is also an operational log simply named *Fuqid.log*, which is off by default. If present this log contains message, primarily intended for debugging.

Experience and Next Steps

The initial project objectives were to determine if we could establish probable cause to obtain search warrants based on *Freenet* activity. We have successfully done that, on both the state and Federal level, albeit on a small scale. We do not yet have enough empirical data to make a definitive conclusion, but it appears in cases where the subject is using *Freenet* we are going to encounter a more technically sophisticated, paranoid subject. We have encountered encryption

in multiple cases, though not thorough enough to prevent us from making a criminal case. We also are finding significant quantities of digital media and equipment. We removed over 20 loose hard drives from one subject's bedroom, in addition to his computers.

In our analysis of the data for Missouri, we notice a mix of activity. We may see the same IP/Port consistently over a period of months. We also frequently see a subject for only a short period of time, maybe a few days or a week. The suspicion is someone with an interest in child abuse materials finds *Freenet* and gives it a try and then either is not in a position to allow it to run long enough to be successful, maybe because of other family members, or becomes frustrated with the relative slowness of *Freenet*. These subjects may still be good targets that are exploring other sources for the material they are seeking.

We also often see an IP address with its port number changing. The port number is set at installation and is not easily changed by the user. It could indicate that there are multiple copies of *Freenet* running behind the IP, or *Freenet* is being removed and reinstalled. In one case we were able to forensically observe that the subject would install *Freenet*, run it for a day or two, and then delete it, only to install it again a few days or weeks later. We have seen a number of cases where the port number appears to change on the fly and we believe this is the result of a NAT router using a different port for outbound traffic.

Of the currently deployed LEF_n nodes, most are connecting to 14-20 peers. The program will allow a maximum of 100 connections, given enough bandwidth. While our efforts have been focused on subjects in the US, the data we collect is worldwide. Observably, roughly two-thirds of the peers we connect to are not in the United States, so opportunities exist for law enforcement in other countries to exploit this data.

The modified *Freenet* program, LEF_n, while successfully obtaining the data we need, should be enhanced in a couple of areas before many more LEF_n nodes are deployed. Currently we have to cycle LEF_n in order to cut off the log file for transmission. Also, the log files contain no source information and this has to be added somewhat manually so we do not lose what nodes created the files. Ideally, forwarding log records to a central server, real-time, would give us the most up-to-date information. We also set the LEF_n nodes data store to the minimum size to avoid facilitating the distribution of contraband blocks. A version that artificially never finds a block in its data store would be best, if the network does not begin to ignore such a node.

The most important need is to make the data we are mining available to a broader law enforcement audience. Today a prototype web interface is available for the databases at the lab in Salem (figure 9). These tools allow a trained investigator to gather enough information to obtain a search warrant. This prototype will need to be migrated to a more robust, production environment, preferably integrated with existing law enforcement systems. Training in what *Freenet* is, how to use it, and how to use the *Black Ice* tools to investigate suspects, is under development.

Search warrants using probable cause based on the Black Ice data will have the same concerns as any P2P search warrant. Did the ISP provide the correct subject in response to our legal demand? Is a neighbor, knowingly or unknowingly, using our subject's Wi-Fi Internet connection? Are there multiple subjects in the target residence? Basic investigative work can help eliminate possible legitimate uses for *Freenet*, such as university environments or businesses with offices in countries with less "open" governments.

Black Ice criminal cases will have to be made on what we find forensically, admissions, and other evidence we find on scene, not on what we saw on the network. Our experience, albeit limited, suggests we will encounter a more complex environment and encryption may be common. While *Freenet* will probably never have the volume of traffic we have seen in Gnutella and other networks, by using the data we collect from *Freenet* we may find suspects that otherwise would never be visible to us. In August of 2013 we executed a search warrant based on Black Ice data where the subject was ultimately found to be not only collecting child abuse material, but was molesting a child victim and videoing this abuse. The only network he was using to obtain child abuse material was *Freenet*. Our window of opportunity may be short; as word gets out the law enforcement is finding *Freenet* CP traders, they will disappear, that is if the developers do not find a way to shut us out first.



The image shows a web interface titled "Black Ice Database Lookups". At the top left is a circular logo with "ICAC" in the center and "INTERNET CHILD ABUSE CENTER" around the perimeter. To the right of the logo is the text "Black Ice" in a stylized, metallic font. Below the header, there are four main sections, each with a title, input fields, and a description:

- Lookup an IP Address**: Includes an input field for "IP Address" and a dropdown menu for "Months" set to "Current-1". Description: "Returns the last time an IP address has been seen by Black Ice and all file-of-interest records for the IP address for the number of months selected."
- Lookup a Port**: Includes input fields for "Port" and "RC", and a dropdown menu for "Months" set to "Current-1". Description: "Returns the last time a Port has been seen by Black Ice and all file-of-interest records for the Port in a region, for the number of months selected. Both Port and RC are required."
- Suspects for a Region**: Includes an input field for "Region Code" and a dropdown menu for "Months" set to "Current-1". Description: "Returns suspect records by file-of-interest for a region (state or province) for the time period requested. Selecting an IP address link will do a IP detail lookup for the same time period."
- US/CA IPs Seen By Black Ice for a Region**: Includes an input field for "Region Code". Description: "Returns all IPs seen by Black Ice for a region, regardless if it has been seen with a file-of-interest."

At the bottom, there is a **Note on Months**: "Current is the current month only, so early in a month it will produce little data. Current-1 includes the current month plus last month. Current-3 will cover a minimum of 90 days (prior 3 months) plus the current month."

Figure 9

Wayne Becker is currently employed by the Dent County Sheriff's Office and is a Special Investigator attached to the Missouri Internet Crimes Against Children (ICAC) Task Force. He investigates cases related to crimes against children, particularly computer related crimes, and the distribution of child abuse material on file sharing systems. He is a certified computer forensic examiner and co-ordinates the South Central Missouri Computer Crimes Task Force. Prior to joining the Sheriff's Office in 2004, Becker was an Information Technology professional in the private sector for 28 years.

He can be reached via email at whbecker@dentcountysheriff.org.

References

- 1) Clarke, "A distributed decentralised information storage and retrieval system," unpublished report, Division of Informatics, University of Edinburgh (1999). Available at <http://www.freenetproject.org/> (2000).
- 2) Clarke, et al., "Freenet: A Distributed Anonymous Information Storage and Retrieval System," in Designing Privacy Enhancing Technologies, Lecture Notes in Computer Science 2009
- 3) O. Sandberg. Distributed routing in small-world networks. In Proceedings of the Eighth Workshop on Algorithm Engineering and Experiments (ALENEX06), 2006.
- 4) O. Sandberg and I. Clarke. The evolution of navigable small-world networks. Technical Report 2007:14, Department of Computer Science and Engineering, Chalmers University of Technology, 2007
- 5) Ian Clarke, Oskar Sandberg, Matthew Toseland, Vilhelm Verendel. Private Communication Through a Network of Trusted Connections: The Dark Freenet, 2006
- 6) <http://www.cs.cornell.edu/home/kleinber/nat00.pdf>
- 7) <http://www.cs.cornell.edu/home/kleinber/swn.pdf>
- 8) <http://en.wikipedia.org/wiki/Freenet>